Before you start the task:
The target of this task is to test your skills in an ansible playbook development and git use. You should focus on a clean code, structure and valid and tested solution. Store all your code in GitHub and use proper commits in your project. Don't forget to document your code.

> **TIME LIMIT:** 1 week

What we want to see:

1. Proper git/GitHub use
   - you should focus on the proper use of git/GitHub
   - whole your project (including playbooks, scripts and documentation) should be in a single repo
   - the repository should be public (you can change it later)
2. Ansible skills
   - project must be solved with an ansible and related tools
   - it's possible to use a script for minor actions but avoid complex solutions
   - choose any structure you consider valid (role, single task, ...)
   - your code must be executable on our machines (RHEL distros mainly)
3. Document our code
   - add proper documentation of your project
   - use comments where needed
   - documentation should be stored in the same repository
4. Testing environment (optional)
   - You can share your test environment settings
   - If you have deployment scripts for your test environment, you can include it

---

TASK

Create an ansible solution that will implement the following changes on multiple servers (2 server min):

1. Deploy LAMP stack on each server and single PHP page
   - use MariaDB as a database
   - use a PHP page of your choice as the main index page (use anything you need, simple solution, we are checking ansible, not a PHP code)
   - page must be reachable from a local web browser after installation
   - page must show the hostname of the machine in some way (e.g. hello from <hostname> page)
   - you don't have to set up a database
2. Deploy users on server based on provided CSV source and setup user login
   - CSV file template can be found lower in this text

    - the name of the file can be different, make sure you find the way to use any CSV file with the correct data
    - deploy users on a host if it is in the CSV file, if not, deploy a generic admin user which has sudo permission (with password)
    - add a comment to the user account in which you mention its owner team
    - if a user is tagged as a functional user, generate a new key pair for that user and store keys on the local machine and the target host, to allow login with the key
    - if the user has its own public RSA key, deploy the key (even if you already created new pair in case of a functional user)
    - server login is allowed only via RSA key, password login is not permitted
    - users without RSA keys should be locked/expired for now (before they provide their keys later)

3. Modify hosts-file

    - modify hosts-file on target servers to know hostnames of all other machines
    - all servers should be reachable from each other by hostname

CSV Example
This is an example of a CSV file. The name can change but there will be always .csv in the name.

Note: Ignore incorrect pub key format, it is just a random string to imitate the line 😃

```
User,Comment,Owner-Team,Functional,Pub-key,
dbuser1,"Database user 1",DB01,1,,
dbuser2,"Database user 1",DB01,1,,
jamesbrown,"ADMINS/UNIX/ID111",ADM,0,"ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAAAQQCo9+B test-a",
emiliagreen,"ADMINS/UNIX/ID222",ADM,0,"ssh-rsa
AAKSKDADNakbdskb2123AKJBDBAKKKADB20 test-b",
webadmin,"Apache2 admin",ADM,1,"ssh-rsa ABBCBABKNALDeapisd232l3eqllaALLALNN
test-c",
```

NOTES
Use the solution you would consider the most valid. We want to see your approach. You might have a chance to explain your choices later. Priority is that the solution is working.

If you can't finish all the tasks, just finish what you are able to. Your code should be working as it is, if something is missing, it should be added later on (avoid unfinished broken code, rather delete it later in your "production code")